Docket No.: 08204/0203518-US0
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Letters Patent of:
Youssri Helmy et al.

Patent No.: 7,286,476

Issued: October 23, 2007

For: ACCELERATING NETWORK PERFORMANCE BY STRIPING AND PARALLELIZATION OF TCP CONNECTIONS

REQUEST FOR CERTIFICATE OF CORRECTION
PURSUANT TO 37 CFR 1.323

Attention: Certificate of Correction Branch
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Upon reviewing the above-identified patent, Patentee noted typographical errors which should be corrected. A listing of the errors to be corrected is attached.

The typographical errors marked with an "A" on the attached list are found in the application as filed by applicant. Payment in the amount of $100.00 covering the fee set forth in 1.20(a) is enclosed.

The errors now sought to be corrected are inadvertent typographical errors the correction of which does not involve new matter or require reexamination.

Transmitted herewith is a proposed Certificate of Correction effecting such corrections. Patentee respectfully solicits the granting of the requested Certificate of Correction.

The Commissioner is authorized to charge any deficiency of up to $300.00 or credit any excess in this fee to Deposit Account No. 04-0100.

Dated: November 7, 2007

Respectfully submitted,

By______________________

Flynn Barrison
Registration No.: 53,970
DARBY & DARBY P.C.
P.O. Box 770
Church Street Station
New York, New York 10008-0770
(212) 527-7700
(212) 527-7701 (Fax)
Attorneys/Agents For Applicant

PTO/SB/44 (04-05)
Approved for use through 04/30/2007. OMB 0651-0033
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.
(Also Form PTO-1050)

UNITED STATES PATENT AND TRADEMARK OFFICE
CERTIFICATE OF CORRECTION

PATENT NO. : 7,286,476

APPLICATION NO.: 10/632,519

ISSUE DATE : October 23, 2007

INVENTOR(S) : Helmy et al.

It is certified that an error appears or errors appear in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

On page 2, in field (56), under "Other Publications", in column 2, line 4, delete "Alorithm" and insert - - Algorithm - -, therefor.

In column 2, line 20, after "need" insert - - to - -.

In column 7, line 23, in Claim 3, after "algorithm" delete "is".

MAILING ADDRESS OF SENDER (Please do not use customer number below):
John W. Branch, Esq.
DARBY & DARBY P.C.

P.O. Box 770
Church Street Station
New York, New York 10008-0770

This collection of information is required by 37 CFR 1.322, 1.323, and 1.324. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1.0 hour to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Attention Certificate of Corrections Branch, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

DARBY & DARBY

Issued Patent Proofing Form **File#: 08204/0203518-US0**
Note: P = PTO Error **A = Applicant Error**

US Serial No.: 10/632,519 **US Patent No.: US 7,286,476 B2** **Issue Dt.: Oct. 23, 2007**
Title: ACCELERATING NETWORK PERFORMANCE BY STRIPING AND PARALLELIZATION OF TCP CONNECTIONS

Sr. No.	P/A	Original		Issued Patent		Description Of Error
		Page	Line	Column	Line	
1	A	Sheet 4 of 4 List of References cited by applicant and considered by examiner (06/15/2005)	Entry 2 Line 1 (Other Documents)	Page 2 Col. 2 (Other Publications)	4	Delete “Alorithm” and insert - - Algorithm - -, therefor.
2	A	Page 3 Specification (08/01/2003)	11	2	20 (Approx.)	After “need” insert - - to - -.
3	A	Page 2 Claims (03/27/2007)	Claim 3 Line 2	7	23	In Claim 3, after “algorithm” delete “is”.

U.S. PATENT DOCUMENTS

2002/0042839	A1 *	4/2002	Peiffer et al.	709/238
2002/0055966	A1 *	5/2002	Border et al.	709/200
2002/0055983	A1 *	5/2002	Goddard	709/217
2002/0071436	A1 *	6/2002	Border et al.	370/395.32
2002/0071438	A1	6/2002	Singh	
2002/0136224	A1	9/2002	Motley	
2003/0123481	A1 *	7/2003	Neale et al.	370/466
2003/0177395	A1	9/2003	Pardee et al.	
2003/0177396	A1 *	9/2003	Bartlett et al.	713/201
2003/0219022	A1 *	11/2003	Dillon et al.	370/395.52
2004/0001519	A1 *	1/2004	Fisher et al.	370/535
2004/0015591	A1 *	1/2004	Wang	709/228
2004/0093420	A1 *	5/2004	Gamble	709/231
2004/0103225	A1 *	5/2004	McAlpine et al.	710/52
2004/0111523	A1 *	6/2004	Hall et al.	709/230
2004/0172475	A1 *	9/2004	Tenereillo et al.	709/229
2004/0215746	A1 *	10/2004	McCanne et al.	709/219
2004/0243703	A1 *	12/2004	Demmer et al.	709/224
2005/0044242	A1 *	2/2005	Stevens et al.	709/228
2005/0243835	A1 *	11/2005	Sharma et al.	370/395.42

OTHER PUBLICATIONS

Huffman, David A., "A Method for the Construction of Minimum-Redundancy Codes," *Proceedings of the I.R.E.*, 40:1098-1101 (1952).

Ziv, Jacob, et al., "A Universal Alorithm for Sequential Data Compression," *IEEE Transactions on Information Theory*, IT23(3):337-343 (1977).

Sivakumar, H. et al. (2000) "PSockets: The Case for Application-level Network Striping for Data Intensive Applications using High Speed Wide Area Networks," Proceedings of Super Computing 2000 (SC2000), Dallas, Texas.

* cited by examiner

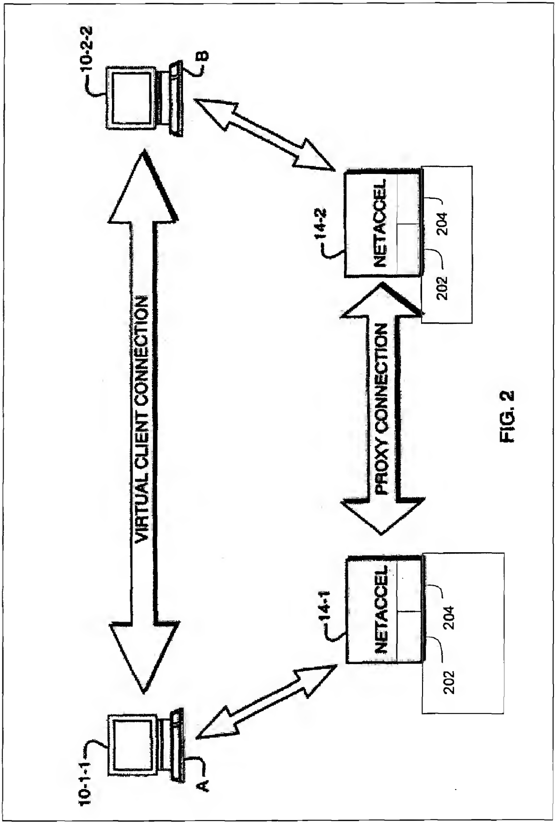
10-2-2
B
VIRTUAL CLIENT CONNECTION
14-2
NETACCEL
204
202
PROXY CONNECTION
14-1
NETACCEL
204
202
10-1-1
A

FIG. 2

ACCELERATING NETWORK PERFORMANCE BY STRIPING AND PARALLELIZATION OF TCP CONNECTIONS

BACKGROUND OF THE INVENTION

The present invention relates to Wide Area Network (WAN) communications, and in particular to placing devices at both ends of a communication link to intercept packets and reducing latency by making parallel transport layer connections.

The growth in data communication traffic, including email, client/server applications, multimedia applications, Internet and intranet applications, has continued to cause critical bandwidth shortages across many networks. This demand for instant data communication often exceeds the available capacity, congestion and delay result. As more software applications move from Local Area Networks (LANs) to Wide Area Networks (WANs), user response time increases, and thus can become a critical limiting factor in smooth operation of an enterprise. With offices widely distributed around the globe and present day budgetary constraints on new telecommunications deployments, the implementation of additional wide area links is cost prohibitive for many international installations. Consequently, network system architects need additional solutions to help them efficiently use existing bandwidth to support more applications and more end users.

The Open Systems Interconnection (OSI) reference model is widely used to define the flow of data traffic across a network. The OSI model has seven layers; each of the seven layers communicates with a layer below it through a specific interface and its peer layer on a different system in the network through a specific protocol. The combination of all networking protocol layers is often referred to as the networking stack. Packet based Transmission Control Protocol over Internet Protocol (TCP/IP) is perhaps the most widely known protocol in use in today's WANs such as the Internet and even private networks. IP is a network layer (Layer 3) protocol that defines a set of standards for addressing and routing of packets across a connectionless network.

The Transmission Control Protocol (TCP) layer is a connection oriented protocol that primarily serves to achieve reliable delivery of data. TCP was originally designed for relatively low-speed, unreliable networks. With the emergence of high-speed WANs, various improvements have been applied to TCP to reduce latency and achieve improved bandwidth. For example, one common way to improve the bandwidth and reduce latency while still using TCP on a reliable high-speed network is to tune the so-called "window size" appropriately.

More particularly, an application typically requests data to be sent to another application at a remote machine. The TCP protocol stack, as typically located in the kernel of an operating system of the sending machine, handles requests from various applications and passes data to the network. This TCP stack partitions the data into segments to be sent over appropriate transmission media, i.e., physical layer connections, and waits for an acknowledgement from the receiving application to know that a particular segment has been received correctly. TCP achieves this by defining how long it will wait, i.e., a window size on both the sender and receiver.

The TCP window size thus defines the amount of time that a sender will wait for an acknowledgement from a receiver before sending any more data. Thus, after sending an initial

block of data once the window size limit is reached, the sender will stop sending data until an acknowledgement from the receiver is returned. It can be appreciated, therefore, that proper selection of TCP window size is potentially critical in improving performance.

In order to achieve maximum performance, it has been suggested to set the window size to approximately the product of the bandwidth times the expected latency delay. When the sender and receiver are connected by a high-speed WAN, this bandwidth-times-delay product value is quite high, and many packets may be sent before a wait state actually occurs.

However, in order to accomplish this improvement in performance, the TCP window size parameter has to be changed at both the source and destination node in order to achieve maximum throughput. This involves changing parameters for the TCP stack in the kernel and typically has to be done by a system administrator at both ends. Also, in order to determine the optimum window size, a performance analysis must often need be done in order to fine-tune its value. This operation can take at least several hours or maybe even several weeks, and it may not deliver the best theoretical results due to fluctuations in network performance due to other application using the same WAN, network congestion, or other factors beyond the sender and receiver's control.

Some have suggested that an application layer process (i.e., at Layer 7) may be used to improve network performance. This is achieved by using a software application at the sending and receiving nodes that implements a technique known as network striping. The striping application partitions data across several open sockets under control of the application program. See, for example, the technique described in "PS Sockets: The Case for Application Level Network Striping or Data Intensive Applications Using High-Speed Wide Area Networks" by Sivakumar, H., et al., Proceedings of Super Computing 2000 (SC2000), Dallas, Tex., November 2000. However, this approach still requires modification of application layer software at both ends of the connection in order to achieve desired performance improvements.

It is also well known to compress data at a much higher layer, such as at the application layer (i.e., at Layer 7). For instance, images can be compressed in a variety of formats such as the Graphics Interchange Format (.gif) or the Joint Photographic Experts Group format (.jpeg). These data file-encoding formats reduce the space required for storage as well as the bandwidth needed for transmission. Hence, at the application layer a server may encode file before transmission to a client on the other end of a WAN connection. Each file received by the client is then decoded at the application layer to generate the original file. However, this approach also requires modifying standard application layer software at each end of the connection.

International Patent Publication Number WO 01/37516 describes a technique whereby a private network connection is implemented between devices known as accelerator exchange servers. A corresponding accelerator exchange client software is installed at a client node. The accelerator exchange client software may be implemented as a browser program adds on, for example.

U.S. Pat. No. 5,657,452 also discloses a method for setting up a connection over a data network. A proxy engine is used to implement data compression. However, this approach requires also installation of the proxy engine so that it runs in a local endpoint node application.

receiving, at the proxy application, intercepted packet traffic;

opening two or more persistent Transmission Control Protocol (TCP) transport layer end-to-end connections in parallel over at least one physical layer connection between the local network accelerator and at least one remote network accelerator, at least the two or more persistent TCP transport layer connections servicing the selected source node; and

striping the transmitting of processed packet traffic to at least one remote network accelerator associated with the destination node which is a destination of the packet traffic via at least the two or more persistent TCP transport layer connections, wherein the remote network accelerator runs another proxy application operating as a proxy for the destination node.

2. A method as in claim 1 wherein a proxy to proxy protocol is employed to specify at least an original TCP transport protocol identifier, original address, and original ports of the nodes.

3. A method as in claim 1 wherein the proxy application uses a dictionary based compression algorithm is to decode the data prior to transmission.

4. A method as in claim 3 wherein a Huffman coding algorithm is applied to compress the data.

5. A method as in claim 3 wherein a dictionary associated with the network layer connection is utilized to service other network layer connections.

6. A data network routing device comprising:

a router, connected to receive incoming packets from a source node, the router examining the incoming packets to determine if they are addressed to a destination node which is not local to the router, and if so, routing them to a socket interface; and

a proxy application, connected to receive incoming packets from the socket interface, the proxy application associated with the router, and the proxy application, acting as a proxy for the source node, also establishing multiple, persistent Transmission Control Protocol (TCP) transport layer end-to-end connections in parallel on behalf of the source node over at least one physical layer connection, the multiple, persistent TCP transport layer connections carrying packets striped in parallel to another proxy application operating as a separate proxy for the destination node.

7. A device as in claim 6 additionally wherein

the proxy application additionally receives packets from a network connection addressed to a destination node which is local to the router.

8. A device as in claim 7 wherein packets are compressed by the proxy application, additionally comprising:

a data decompressor, for decompressing packets so received; and

wherein the router also forwards decompressed packets to the destination node.

9. A device as in claim 6 wherein at least one TCP transport layer sessions are carried over a persistent connection established with another data network routing device having the other proxy application running thereon.

10. A device as in claim 6 wherein a proxy to proxy protocol is used to pass original source node and destination node information.

11. A device as in claim 6 wherein a proxy to proxy protocol specifies an original protocol type for the packets.

12. A method for communicating a data stream between a source node and a destination node, comprising:

establishing a plurality of persistent Transmission Control Protocol (TCP) transport layer connections in parallel between a first proxy application in communication with the source node and a second proxy application in communication with the destination node, wherein the plurality of persistent TCP transport layer connections are provided between the first proxy application and the second proxy application over at least one physical layer connection;

enabling the first proxy application to forward the data stream from the source node to the destination node, wherein the data stream is provided over a first TCP transport layer client connection to the first proxy application, and wherein the first proxy application stripes the data stream over the plurality of persistent TCP transport layer connections between the first proxy application and the second proxy application; and

enabling the second proxy application to provide the striped data stream received over the plurality of persistent TCP transport layer connections to the destination node over a second TCP transport layer client connection.

13. The method of claim 12, further comprising:

enabling the communication of the data stream to the destination node to appear to the source node as occurring directly over the first TCP transport layer client connection; and

enabling communication with the source node to appear to the destination node as occurring directly over the second TCP transport layer client connection.

14. The method of claim 12, wherein enabling the first proxy application to receive the data stream for the destination node, further comprises enabling the first proxy application to spoof an address associated with the source node for the first TCP transport layer client connection.

15. The method of claim 12, wherein enabling the second proxy application to provide the data stream to the destination node, further comprises enabling the second proxy application to spoof at least another address associated with the destination node for the second TCP transport layer client connection.

16. The method of claim 12, further comprising if the destination node is remote to the source node, providing the data stream to the first proxy application over a socket interface.

17. The method of claim 12, wherein if a plurality of data streams are provided over the plurality of persistent TCP transport layer connections, enabling the first proxy application to employ a dictionary based compression algorithm on at least a portion of at least one data stream that is communicated to the second proxy application.

18. The method of claim 12, further comprising communicating the striped data stream over a plurality of physical layer connections, wherein at least a portion of the plurality of persistent TCP transport layer connections are provided over each of the plurality of physical layer connections.

19. A system for enabling communication of a data stream between a source node and a destination node, comprising:

a first proxy application in communication with the source node;

a second proxy application in communication with the destination node,